

# Cómo Construir una base de datos

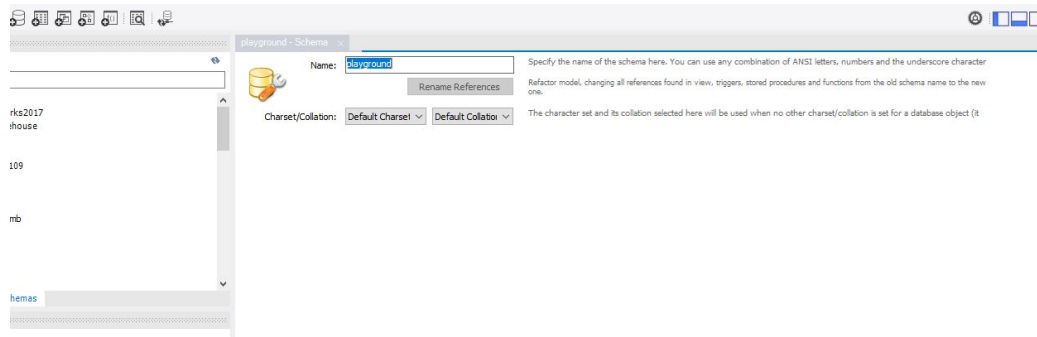
**DigitalHouse** >  
Coding School



**Certified Tech  
Developer**  
The Ultimate Degree

# ¡Empecemos!

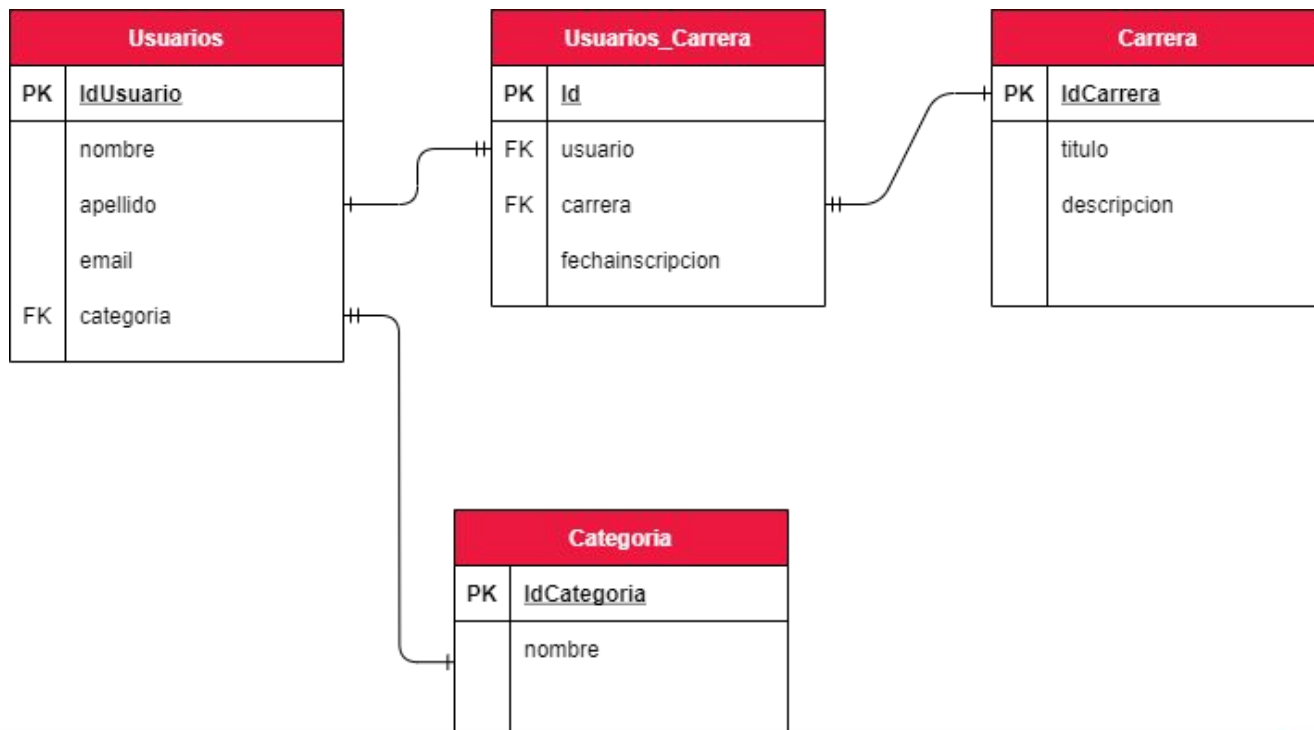
Tomando como base el ejercicio de DER Playground realizado, vamos a realizar el paso a paso para **crear una parte de la base de datos:**



SQL

```
CREATE SCHEMA playground ;
```

# DER - Playground



# Ejemplo CREATE TABLE “categorias”

SQL

```
CREATE TABLE playground.categorias (  
    idcategoria INT NOT NULL,  
    nombre VARCHAR(100) NULL,  
    PRIMARY KEY (idcategoria));
```






# Ejemplo CREATE TABLE “usuarios”

usuarios - Table

Table Name:  Schema: **playground**

Charset/Collation:   Engine:

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 idusuario	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 nombre	VARCHAR(100)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 apellido	VARCHAR(100)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 email	VARCHAR(50)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 categoria	INT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Column Name:

CharSet/Collation:

Comments:

Data Type:

Default:

Storage: ☐ Virtual ☐ Stored

☐ Primary Key ☐ Not Null ☐ Unique

☐ Binary ☐ Unsigned ☐ Zero Fill

☐ Auto Increment ☐ Generated

Columns Indexes Foreign Keys Triggers Partitioning Options

Apply Revert

SQL

```
CREATE TABLE playground.usuarios (  
    idusuario INT NOT NULL,  
    nombre VARCHAR(100) NULL,  
    apellido VARCHAR(100) NULL,  
    email VARCHAR(50) NULL,  
    categoria INT NULL,  
    PRIMARY KEY (idusuario),  
    FOREIGN KEY (categoria) REFERENCES playground.categorias  
    (idcategoria) );
```

# Ejemplo CREATE TABLE “carrera”

SQL

```
CREATE TABLE playground.carrera (  
  idcarrera INT NOT NULL,  
  titulo VARCHAR(45) NULL,  
  descripcion VARCHAR(100) NULL,  
  PRIMARY KEY (idcarrera));
```

## Ejemplo CREATE TABLE “usuarios\_carrera”

SQL

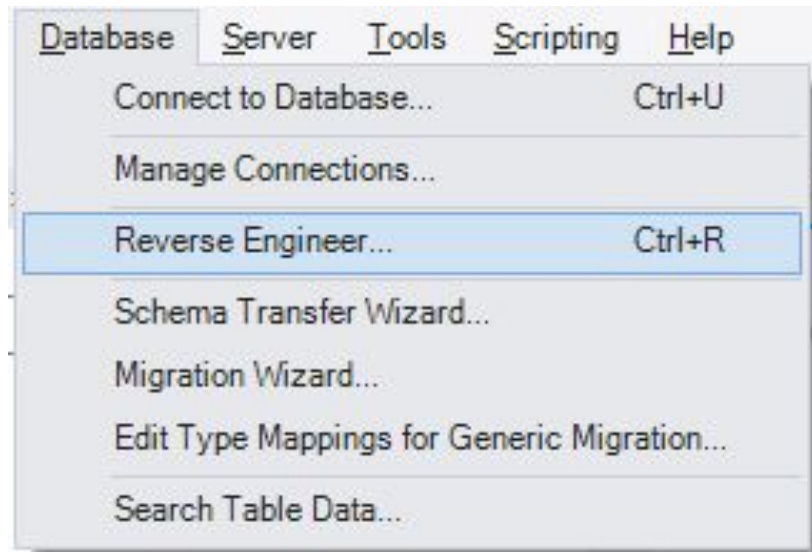
```
CREATE TABLE playground.usuarios_carrera (  
  id INT NOT NULL,  
  usuario INT NULL,  
  carrera INT NULL,  
  fechainscripcion DATE NULL,  
  PRIMARY KEY (id),  
  FOREIGN KEY (usuario)  
  REFERENCES playground.usuarios (idusuario),  
  FOREIGN KEY (carrera)  
  REFERENCES playground.carrera (idcarrera));
```



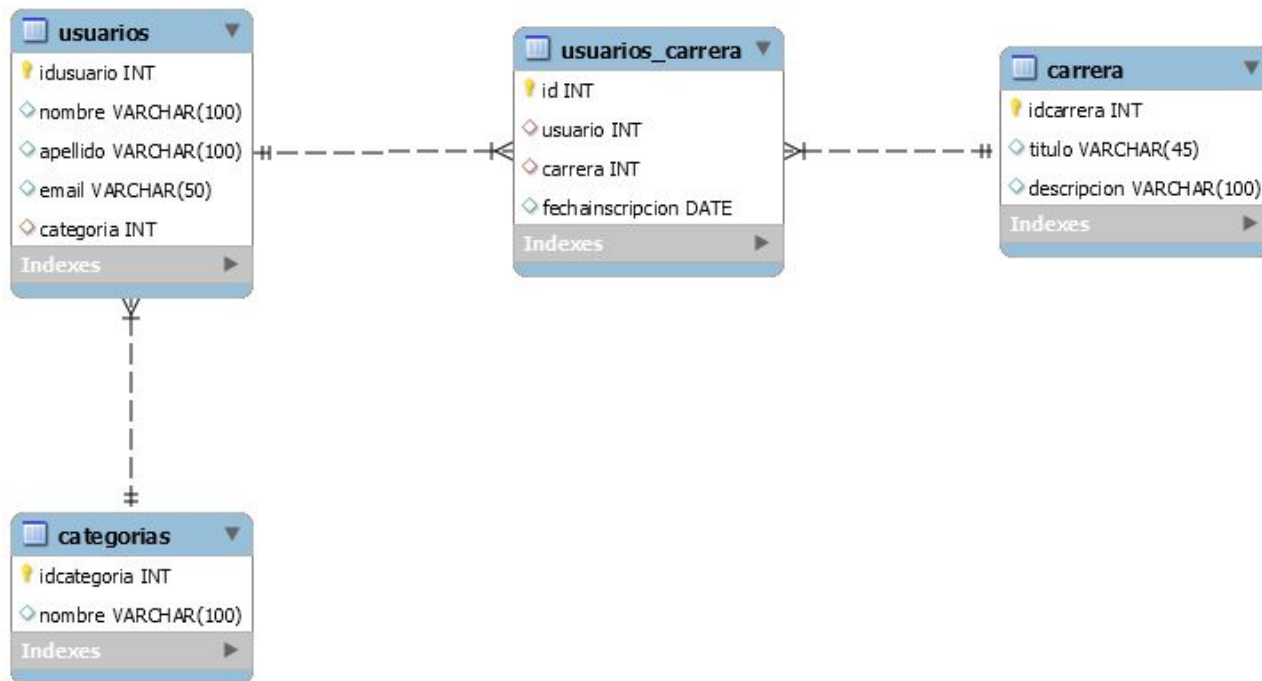
# Validar el modelo creado

Continuamos con las tablas de **carrera** y de **Usuarios\_Carrera** con las claves foráneas a las tablas de **usuarios** y **carrera**.

Luego, podemos realizar ingeniería inversa para controlar que el modelo relacional es el correcto.



# WORKBENCH - DER - Playground



# Insertar datos

Vamos a insertar datos en las tablas:

- Categorías: “Alumno”, “Docente”, “Editor” y “Administrador”.
- Usuario: “Juan Perez [jperez@gmail.com](mailto:jperez@gmail.com) categoria alumno”.
- Carrera: “Certified Tech Developer Carrera de programación y desarrollo de productos digitales”.
- Juan se inscribió el 1/3/2021 a CTD.



# Ejemplo INSERT - Categorías

SQL

```
INSERT INTO playground.categorias
(idcategoria, nombre)
VALUES
(1, "Alumno"),
(2, "Docente"),
(3, "Editor"),
(4, "Administrador");
```

Desde la parte de administración nos avisan que no existe más la categoría "Editor".

¿Qué tenemos que hacer?

SQL

```
DELETE FROM playground.categorias  
WHERE nombre = "Editor";
```

Ahora, ¿Qué sucede si intentamos eliminar la categoría “Alumno”?

SQL

```
DELETE FROM playground.categorias  
WHERE nombre = "Alumno";
```



Nos va a generar un error similar a *“Cannot delete or update a parent row: a foreign key constraint fails”*.  
Esto indica que no se puede eliminar la categoría “Alumno” dado que debe haber algún alumno bajo esta categoría.

DigitalHouse>  
Coding School